Ye Zee News

Registered By Australian Post
Publication No. SBG0711

Issue #4
October 1984

Return Address
30 Seventh ave. Cheltenham S.A. 5014

This issue sees the first hardware contribution in print. It is an easy to do add on keyboard which gives the VZ a realy profesional feel. Other hardware mods either ready for publication or in preparation are Joystick interface, RAM expansion, general purpose I/O port and a voice synthesiser.

On the software side we have a few useful utility programs, a group of programs to do simple electronic calculations and a program to tell world times, with a high resolution graphics map of the world.

Ron Carson's second machine code tutorial coincides with the release of DSE's Editor Assembler software. This software (or something similar) is a must if you want to follow Ron's tutorials and try and pick up machine code programing.

Jah

A few people have pointed out that last months Geography program was difficult to enter because there was no indication of how many graphics characters to put on each line.

This is a valid complaint and I will try to avoid the problem in future. To do this I need your help, if you are contributing programs to Ye Zee News please use REM lines to tell the reader how many graphics characters or spaces should be used.

Tandy sell pens suitable for the VZ 200 printer/plotter for $4.95 per pack of three. Either 3 black or 3 colours (red, green & blue). This is a bit cheaper than DSE but, since most people use more black than coloured pens the Tandy packs make a bit more sense.

```
10 REM PERFORATION SKIP FOR FANFOLD PAPER
20 CLS:PRINT:PRINT"    PERFORATION SKIP FOR FANFOLD PAPER":PRINT:PRINT
30 PRINT"    AFTER RUNNING THIS PROGRAM  CLOAD YOUR PROGRAM & ";
40 PRINT"LLIST IT.  THIS ROUTINE WILL LIST 60 LINES,GIVE 6";
50 PRINT" LINE FEEDS, LIST 60 LINESETC. THE LINE COUNTER";
60 PRINT" MUST BE   SET TO ZERO AFTER EACH LISTING  BY A POKE ";
70 PRINT"30761,0":PRINT:PRINT:PRINT
100 FORM=-20480TO-20447:READA:POKEM,A:NEXT:POKE31200,0
110 POKE31201,176:POKE30761,0:POKE31199,205
120 DATA58,41,120,254,60,192,205,254,32,58,33,176,61,50,33,176
130 DATA202,22,176,195,6,176,62,0,50,41,120,62,6,50,33,176,201,6
```

```
B000:3A2978    LD   A,(7829)
B003:FE3C      CP   3C
B005:C0        RET  NZ
B006:CDFE20    CALL 20FE
B009:3A21B0    LD   A,(B021)
B00C:3D        DEC  A
B00D:3221B0    LD   (B021),A
B010:CA16B0    JP   Z,B016
B013:C306B0    JP   B006
B016:3E00      LD   A,00
B018:322978    LD   (7829),A
B01B:3E06      LD   A,06
B01D:3221B0    LD   (B021),A
B020:C9        RET
```

Page skip for fanfold paper
by B Hubble

This program may be of interest to readers who own GP100 or similar printers. Although the VZ has a printer line counter it appears to serve no function in the LLIST mode.

This program will list 60 lines per page then leave a six line gap to allow for the paper fold. This makes it easier to file, read etc long listings.

To save paper if you want to test the program just POKE 30761,56 for example and it will list 4 lines and then 6 line feeds.

ELECTRONICS UTILITY

By P. Boyce

```
30 ' # RESISTORS IN PARRALLEL #
40 CLS
45 FORS=1TO30
50 PRINT@128,"THIS PROGRAM IS FOR CALCUL
ATING"
55 PRINT@161,"THE TOTAL RESISTANCE 'RT'
OF"
60 PRINT@194,"TWO KNOWN RESISTOR VALUES"
65 NEXTS
90 CLS
95 PRINT@3,"# RESISTORS IN PARALLEL #"
110 PRINT@33,"SELECT TWO RESISTANCE VALU
ES"
115 REM THE WORDS 'IN' 'OHMS' AND 'RT' A
RE IN INVERSE MODE
120 PRINT@72,"IN OHMS FOR RT"
140 REM SHIFT "K" AND "M"
150 Y$="--/\/\/\/\/\-- IS"
160 H$="OHMS"
180 PRINT" ENTER VALUE OF FIRST RESISTOR
";FR$
200 INPUT X
220 PRINT@290,"R1";Y$;X;H$
240 PRINT@160," ENTER VALUE OF SECOND RE
SISTOR";SR$
260 INPUT Y
280 PRINT@386,"R2";Y$;Y;H$
300 P=X*Y/(X+Y)
320 PRINT@325,"NOW CALCULATING 'RT'"
340 FORC=1TO1000:NEXTC
360 PRINT@322,"THE PARALLEL RESISTANCE F
OR"
380 PRINT@358,"RT IS";P;H$
400 REM 'RT' IS IN INVERSE MODE
420 FORL=1TO5000:NEXTL:GOTO90

440 '# RESISTOR/ZENER NETWORK #
450 CLS:PRINT
460  PRINT"  SELECT ZENER VOLTAGE FROM
     PREFERED VALUES"
470 PRINT" NUMBERS ONLY PLEASE"
480 PRINT" INPUT VOLTAGE"
490 INPUT V1
500 PRINT" OUTPUT VOLTAGE,(ZENER VALUE)"
510 INPUT V2
520 PRINT" CURRENT DRAW IN AMPS"
530 PRINT" E.G. .05A=50MA  , .45A=450MA"
540 INPUT I
550 P2=I*V2
560 V3=V1-V2
570 R=V3/I
580 P1=I*V3
590 PRINT" RESISTOR IS ";R"OHMS @";P1;"
    WATT/S"
600 PRINT" ZENER IS";P2"WATT/S"
610 FORL=1 TO 8000:NEXTL:GOTO 440
620 END
680 '# RESISTOR COLOUR DECODER #
700 CLS
710 PRINT:PRINT"RESISTOR COLOUR CODE TRA
NSLATOR"
720 PRINT"ENTER THE THREE COLOURS FROM
  LEFT TO RIGHT,SEPARA";
730 PRINT"TING THEM   BY A COMMA (,)":IN
PUT M$,S$,F$
740 PRINT:PRINT"NOW THE FORTH COLOUR,IF
NONE,"
750 INPUT"THEN ENTER 'NO COLOUR'";T$
760 IFM$="BLACK"THEN M=00
770 IFM$="BROWN"THEN M=10
780 IFM$="RED"THEN M=20
790 IFM$="ORANGE"THEN M=30
800 IFM$="YELLOW"THEN M=40
810 IFM$="GREEN"THEN M=50
820 IFM$="BLUE"THEN M=60
830 IFM$="VIOLET"THEN M=70
840 IFM$="GREY"THEN M=80
850 IFM$="WHITE"THEN M=90
860 IFS$="BLACK"THEN S=0
870 IFS$="BROWN"THEN S=1
880 IFS$="RED"THEN S=2
890 IFS$="ORANGE"THEN S=3
900 IFS$="YELLOW"THEN S=4
910 IFS$="GREEN"THEN S=5
920 IFS$="BLUE"THEN S=6
930 IFS$="VIOLET"THEN S=7
940 IFS$="GREY"THEN S=8
950 IFS$="WHITE"THEN S=9
960 IFF$="GOLD"THEN F=.1
970 IFF$="BLACK"THEN F=1
980 IFF$="BROWN"THEN F=10
990 IFF$="RED"THEN F=100
1000 IFF$="ORANGE"THEN F=1000
1010 IFF$="YELLOW"THEN F=10000
1020 IFF$="GREEN"THEN F=100000
1030 IFF$="BLUE"THEN F=1000000
1040 IFF$="VIOLET"THEN F=10000000
1050 IFF$="GREY"THEN F=100000000
1060 IFF$="WHITE"THEN F=1000000000
1070 IFT$="GOLD"THEN P$="5%"
1080 IFT$="SILVER"THEN P$="10%"
1090 IFT$="NO COLOUR"THEN P$="20%"
1100 R=(M+S)*F
1110 PRINT" RESISTANCE=";R;"OHMS"
1120 PRINT" TOLERANCE= ";P$
1130 RI=((M+S)*F)/1000:PRINT" RESISTANCE
=";RI;"KILOHMS"
1140 PRINT" TOLERANCE= ";P$
1150 PRINT:INPUT"DO YOU HAVE ANY MORE PR
OBLEMS";Z$
1160 IFZ$="N"THEN END ELSE RUN 700
```

From Colin Hudson

To get the TP-40 Printer/Plotter to do a reverse line feed use,

```
OUT 12,11 : OUT 13,0
```

Do this with care as the VZ does not check to see if the printer buffer is full

Last month I gave a few examples of using the extra commands, for example

```
1 PRINT!(32,"*")
POKE 31470,196
RUN
```

Some readers would like to print things at a specific screen location with a PRINT@, this can be done as follows,

```
1 PRINT@96,!(32,"*")
POKE 31474,196
RUN
```

This time you POKE at 31474 because the @96, displaces the ! 4 places in the RAM. The same thing can be done with any of the

Correction to ON ERROR command

Colin Hudson wrote to me after last months issue to explain that I had "got it wrong". The correct method for implementing ON ERROR by Colin's technique is described here, this time I'll quote from his letter and perhaps not muck it up again.

> The bytes 30960,1 must be POKE'd with the RAM address of the line not the BASIC line number. I have included two programs to find the RAM address of a line, one using BASIC and the other using machine code. To use the BASIC routine all that is necessary is to set the variable DL to the BASIC line number. ON ERROR GOTO can then be simulated by using line # 210 in the BASIC listing. Another use for the routine is the simulation of RESTORE (line number), this is done by line # 310. The second BASIC program is the data statements and a sample way to use the machine code program. In the example DLx must be set to the line number and the USR routine finds the line address. The variable Fx is used as a flag. In the example RESTORE (line number) is simulated. It can be used for ON ERROR by omitting the X=X-1 in line 60 and POKE'ing 30960,1 instead of the DATA pointer 30975,6 in line 79.

BASIC

```
10 REM SUB-ROUTINE TO FIND THE START ADD
RESS OF A LINE
20 DF=1: DP=PEEK(30884)+256*PEEK(30885)
30 FORDW=0TO3: DB=DP+DW: IFDB>32767THEN
DB=DB-65536
40 DT(DW)=PEEK(DB): NEXT
50 DT=DT(0)+256*DT(1): DN=DT(2)+256*DT(3
)
60 IFDN<DL AND DT>DPTHEN DP=DT:GOTO30
70 IFDL<>DN OR DT<DP,DF=-1: REM LINE NOT
 FOUND
80 DZ=DP-256*INT(DP/256):DY=INT(DP/256):
RETURN


200 REM TO SET UP ON ERROR GOTO
210 IFDF=1THENPOKE30960,DZ:POKE30961,DY


300 REM TO SET UP DATA POINTER
310 IFDF=1, POKE30975,DZ-1:POKE30976,DY
```

MACHINE CODE DISASSEMBLY TO FIND THE RAM ADDRESS OF A BASIC LINE

```
Z80 Z-80 DISASSEMBLER V 1.0

CF08:00          NOP                    ?
CF09:2AA478      LD   HL,(78A4)         *?x
CF0C:44          LD   B,H               D
CF0D:4D          LD   C,L               M
CF0E:23          INC  HL                #
CF0F:23          INC  HL                #
CF10:5E          LD   E,(HL)            ^
CF11:23          INC  HL                #
CF12:56          LD   D,(HL)            V
CF13:EB          EX   DE,HL             ?
CF14:ED5B2179    LD   DE,(7921)         ?[!y
CF18:A7          AND  A                 ?
CF19:ED52        SBC  HL,DE             ?R
CF1B:280A        JR   Z,0A   (CF27)     (?
CF1D:D0          RET  NC                ?
CF1E:60          LD   H,B               `
CF1F:69          LD   L,C               i
CF20:4E          LD   C,(HL)            N
CF21:23          INC  HL                #
CF22:46          LD   B,(HL)            F
CF23:60          LD   H,B               `
CF24:69          LD   L,C               i
CF25:18E7        JR   E7     (CF0E)     ??
CF27:ED432179    LD   (7921),BC         ?C!y
CF2B:C9          RET                    ?

CF2C:C9          RET                    ?

CF2D:C9          RET                    ?


D O N E

INSTRUCTIONS:              27
BYTES:                     38
```

```
DECIMAL DATA STATEMENTS AND SAMPLE ROUTI
NE TO USE USR
10 DATA0,42,164,120,68,77,35,35,94,35,86
,235,237,91,33
20 DATA121,167,237,82,40,10,208,96,105,7
8,35,70,96,105,24,231
30 DATA237,67,33,121,201,201,201
35 REM SET UP USR ROUTINE
40 FORW=0TO37:READA:POKE53000+W-65536,A:
NEXT
45 ' SET UP DLx AND MACHINE CODE POINTER
S AND CALL USR
50 INPUT"ENTER LINE";DLx:POKE30863,207:P
OKE30862,8:X=USR(DLx)
55 ' CHECK IF THE LINE IS FOUND YES X=X-
1 TO SET UP DATA POINTER
56 ' SET FLAG TO 1 PRINT ERROR MESSAGE
60 C=256:IFX<>DLx,X=X-1:Fx=-1ELSEPRINT"L
INE NOT FOUND":Fx=1:RETURN
65 ' CHECK FLAG THEN SET UP DATA POINTER
 BYTES
70 IFFx,POKE30975,X-C*INT(X/C):POKE30976
,INT(X/C)
80 RETURN
```

WORLD TIMES

```
1 CLEAR 1000:DIME$(36):GOSUB 900:CLS
2 REM R.G.DAVIS 8 AVOCARDO ST   MILDURA 3500
4 PRINT'  WORLD TIMES     BY R.G.DAVIS':PRINT:PRINT
6 PRINT'ENTER YOUR LOCAL TIME'
8 PRINT'IN 24 HOUR MODE EG..22,36':PRINT
11 INPUT H,M
12 Z=H-10:O=M
14 GOSUB 320:GH=Z:GM=O
15 MODE(1):GOSUB500
16 Y1=59
21 Z=GH:O=GM
30 E$='LONDON':C1=2:C2=4:X3=61:Y3=22:GOSUB110
36 E$='MOSCOW':Z=GH+2:O=GM+30:C1=2:C2=4:X3=74:Y3=21:GOSUB110
40 E$='MILDURA':Z=GH+10:O=GM:C1=2:C2=3:X3=112:Y3=49:GOSUB 110
50 E$='LOS ANGELES':Z=GH-8:C1=4:C2=2:X3=18:Y3=29:GOSUB 110
60 E$='NEW YORK':Z=GH-5:O=GM:C1=4:C2=2:X3=34:Y3=28:GOSUB 110
70 E$='RIO DE JANEIRO':Z=GH-3:C1=2:C2=3:X3=46:Y3=45:GOSUB110
100 GOTO 21
105 REM DISPLAY NUMBERS
110 X1=0:GOSUB320:C$=STR$(Z)
120 GOSUB130
121 C$=STR$(O):X1=X1+4
122 GOSUB130
125 GOTO 200
130 D=LEN(C$)
131 IFD=2THEN A$(0)=E$(0)ELSEGOTO135
132 N=0:GOSUB 1010:N=1:Y1=59:GOSUB1100:X1=X1+4
135 FORU=2TOD
140 E=VAL(MID$(C$,U,1))
150 N=0:A$(0)=E$(E):GOSUB1010:N=1:GOSUB1100
160 X1=X1+4:NEXT:RETURN
200 X1=32:REM DISPLAY LETTERS
212 L=LEN(E$)
220 FORU=1TOL
221 L$=MID$(E$,U,1)
222 Q=ASC(L$):Q=Q-55
223 IF Q<0THEN X1=X1+4:GOTO240
230 N=0:A$(0)=E$(Q):GOSUB1010:N=1:Y1=59:GOSUB1100:X1=X1+4
235 IFQ=22 OR Q=32 THENX1=X1+2
240 NEXT
245 REM TURN PIXEL ON AND OFF
250 FORI=1TO10
252 COLORC1:SET(X3,Y3)
254 FORN=1TO100:NEXT:SOUND1,2
256 RESET(X3,Y3)
258 FORN=1TO100:NEXT
260 NEXT:COLORC2:SET(X3,Y3)
265 REM RUB OUT NAMES OF CITY
270 COLOR1:Y=59:FORI=1TO5
272 FORN=21TO95
274 SET(N,Y):NEXT
276 Y=Y+1:NEXT
278 RETURN
315 REM CHECK FOR ILEGAL TIMES
320 IF O>59 THEN O=O-60:Z=Z+1
322 IF O<0 THEN O=O+60:Z=Z-1
324 IF Z<0 THEN Z=Z+24
325 IF Z>23 THEN Z=Z-24
326 RETURN
490 REM DISPLAY MAP
500 FORI=28672 TO 30505:READ A:POKEI,A
505 NEXT
510 RETURN
```

* Note that all ' in this listing should be entered as " and that all lines from 910 to 935 should end in "

```
660 DATA 0,0,0,0,1,85,85,85,80,0,0,0,0,0,0,21,64,0,15,255,255
661 DATA 255,255,255,255,255,12,252,0,0,0,0
662 DATA 0,0,0,0,0,85,85,85,64,0,0,0,0,0,5,85,84,20,15
663 DATA 255,255,255,255,255,255,255,3,0,0,0,0,0
664 DATA 0,0,0,0,0,21,85,0,64,0,0,0,0,0,21,85,85,85,127,252
665 DATA 255,255,255,255,255,255,0,0,0,0,0,0
666 DATA 0,0,0,0,0,15,252,0,0,0,0,0,0,0,85,85,85,85,79,255,207
667 DATA 255,255,255,255,255,0,0,0,0,0,0
668 DATA 0,16,0,0,0,0,252,12,0,0,0,0,0,0,85,85,85,85,83,255,240
669 DATA 15,255,15,255,192,0,0,0,0,0,0
670 DATA 0,4,0,0,0,0,63,252,0,0,0,0,0,1,85,85,85,85,84,255,240
671 DATA 3,252,3,252,0,192,0,0,0,0,0
672 DATA 0,0,0,0,0,0,0,63,0,0,0,0,0,1,85,85,85,85,85,60,0,3,240
673 DATA 0,255,0,192,0,0,0,0,0
674 DATA 0,0,0,0,0,0,0,3,2,160,0,0,0,0,85,85,85,85,85,68,0,0
675 DATA 240,0,207,0,0,0,0,0,0,0
676 DATA 0,0,0,0,0,0,0,3,250,170,128,0,0,0,21,85,85,85,85,84,0
677 DATA 0,240,0,192,0,48,0,0,0,0,0
678 DATA 0,0,0,0,0,0,0,0,10,170,170,0,0,0,0,0,85,85,85,80,0,0,0
679 DATA 1,48,20,0,0,0,0,0,0
680 DATA 0,0,0,0,0,0,0,0,42,170,170,128,0,0,0,0,21,85,85,64
681 DATA 0,0,0,0,16,1,64,87,0,0,0,0
682 DATA 0,0,0,0,0,0,0,0,42,170,170,170,128,0,0,0,21,85,84
683 DATA 0,0,0,0,0,16,1,64,87,0,0,0,0
684 DATA 0,0,0,0,0,0,0,0,42,170,170,170,128,0,0,0,5,85,85
685 DATA 0,0,0,0,0,0,1,80,0,3,240,0,0,0
686 DATA 0,0,0,0,0,0,0,0,10,170,170,170,128,0,0,0,5,85,85
687 DATA 0,0,0,0,0,0,0,0,0,12,0,0,0
688 DATA 0,0,0,0,0,0,0,0,2,170,170,170,0,0,0,0,5,85,85,12,0,0
689 DATA 0,0,0,0,2,130,0,0,0,0
690 DATA 0,0,0,0,0,0,0,0,0,170,170,168,0,0,0,0,5,85,84,60,0,0,0
691 DATA 0,0,0,42,162,128,0,0,0
692 DATA 0,0,0,0,0,0,0,0,0,42,170,168,0,0,0,0,5,85,80,240
693 DATA 0,0,0,0,0,2,170,170,160,0,0,0
694 DATA 0,0,0,0,0,0,0,0,0,42,170,128,0,0,0,0,5,85,80,240
695 DATA 0,0,0,0,0,10,170,170,168,0,0,0
696 DATA 0,0,0,0,0,0,0,0,0,42,170,0,0,0,0,0,1,85,64,0,0,0,0,0,0
697 DATA 42,170,170,168,0,0,0
698 DATA 0,0,0,0,0,0,0,0,0,170,168,0,0,0,0,0,1,85,0,0,0,0,0,0,0
699 DATA 42,170,170,168,0,0,0
700 DATA 0,0,0,0,0,0,0,0,0,170,168,0,0,0,0,0,0,84,0,0,0,0,0,0,0
701 DATA 10,0,42,168,0,12,0
702 DATA 0,0,0,0,0,0,0,0,0,170,128,0,0,0,0,0,0,0,0,0,0,0,0,0,0
703 DATA 0,0,2,160,0,3,0
704 DATA 0,0,0,0,0,0,0,0,2,170,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
705 DATA 0,0,0,0,3,0
706 DATA 0,0,0,0,0,0,0,0,2,160,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
707 DATA 0,0,240,0,60,0
708 DATA 0,0,0,0,0,0,0,0,2,160,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
709 DATA 0,0,0,0,240,0
710 DATA 0,0,0,0,0,0,0,0,2,160,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
711 DATA 0,0,0,0,0,0
712 DATA 0,0,0,0,0,0,0,0,2,128,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
713 DATA 0,0,0,0,0,0
714 DATA 0,0,0,0,0,0,0,0,0,128,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
715 DATA 0,0,0,0,0,0
716 DATA 0,0,0,0,0,0,0,0,0,32
```

```
590 REM DATA FOR MAP
600 DATA 0,0,0,0,0,0,0,0,0,0,3,255,255,255,255,0,0,0,0,0,0,0,0,0,0
601 DATA 0,0,0,0,0,0,0,0,0
602 DATA 0,0,0,0,0,0,0,0,0,0,3,255,255,255,255,0,0,0,0,0,0,0,0,0
603 DATA 0,0,0,0,0,0,0,0,0
604 DATA 0,0,0,0,0,0,0,0,0,0,15,255,255,255,255,0,0,0,0,0,0,0,0,0
605 DATA 0,0,0,0,0,0,0,0,0
606 DATA 0,0,0,0,0,0,0,0,0,0,255,255,255,255,252,0,0,0,0,0,0,0,0,0
607 DATA 0,0,0,0,0,0,0,0,0
608 DATA 0,0,0,0,0,0,0,0,0,0,63,255,255,255,252,0,0,0,0,0,0,0,0,0
609 DATA 0,0,0,0,0,0,0,0,0
610 DATA 0,0,0,0,0,0,0,0,0,0,63,255,255,255,252,0,0,0,0,0,0,0,0,0
611 DATA 0,60,0,0,0,0,0,0,0
612 DATA 0,0,0,0,0,0,0,0,0,0,15,255,255,255,252,0,0,0,0,0,0,0,0,240
613 DATA 0,0,63,240,0,0,0,0,0,0
614 DATA 0,0,0,0,0,0,0,0,0,0,0,63,255,255,252,0,0,0,0,0,0,0,63,0,0
615 DATA 63,255,240,0,0,0,0,0,0
616 DATA 0,0,0,0,12,0,0,0,0,0,63,255,255,252,0,0,0,0,0,0,0,60,0,0
617 DATA 255,255,192,0,0,0,0,0,0
618 DATA 0,0,0,0,15,0,0,15,0,0,15,255,255,240,0,0,0,0,0,0,0,48,0,3
619 DATA 255,255,252,0,0,0,0,0,0
620 DATA 0,0,0,0,60,243,192,15,252,0,15,255,255,240,0,0,0,0,0,0,0
621 DATA 240,15,15,255,255,255,240,3,240,0,0,0
622 DATA 0,5,0,0,0,255,195,207,255,0,15,255,255,240,0,0,0,0,0,0,0
623 DATA 48,63,255,255,255,255,255,255,252,0,0,0
624 DATA 0,85,84,3,240,63,240,192,15,192,3,255,255,240,0,0,0,0,240
625 DATA 192,0,0,15,255,255,255,255,255,255,240,0,0
626 DATA 1,85,85,255,255,192,0,243,3,240,3,255,255,0,0,0,0,0,255
627 DATA 252,48,15,207,255,255,255,255,255,255,255,255,255,63
628 DATA 0,21,85,255,255,195,255,255,3,252,3,255,240,0,0,0,3,255
629 DATA 255,51,255,255,255,255,255,255,255,255,255,255,255,252
630 DATA 1,85,85,255,255,255,255,252,15,204,3,255,0,51,192,0,15
631 DATA 243,240,63,255,255,255,255,255,255,255,255,255,255
632 DATA 255,255
633 DATA 0,21,85,255,255,255,255,240,0,240,3,255,0,15,0,0,63,195
634 DATA 255,255,255,255,255,255,255,255,255,255,255,255,255,12
635 DATA 1,85,85,255,255,255,255,240,12,48,0,252,0,0,0,0,255,15
636 DATA 255,255,255,255,255,255,255,255,255,255,255,255,255,0
637 DATA 1,85,65,255,255,255,255,0,15,192,0,60,0,0,0,3,255,63
638 DATA 255,255,255,255,255,255,255,255,255,255,255,207,240,0
639 DATA 0,4,0,63,255,255,255,0,15,204,0,0,0,0,0,3,207,15,255
640 DATA 255,255,255,255,255,255,255,255,252,63,12,0,0
641 DATA 0,16,0,3,255,255,255,192,15,255,0,0,0,0,0,192,15,63,255
642 DATA 255,255,255,255,255,255,255,255,240,0,60,0,0
643 DATA 0,64,0,0,255,255,255,255,15,255,192,0,0,0,12,240,192,63
644 DATA 255,255,255,255,255,255,255,255,255,192,0,240,0,0
645 DATA 0,0,0,0,255,255,255,255,63,255,192,0,0,0,12,48,255,255
646 DATA 255,255,255,255,255,255,255,255,255,252,192,240,0,0
647 DATA 0,0,0,0,63,255,255,255,255,255,0,0,0,0,3,195,255,255
648 DATA 255,255,255,255,255,255,255,255,255,252,192,0,0,0
649 DATA 0,0,0,0,5,85,85,85,255,240,0,0,0,0,0,0,63,255,255,255
650 DATA 255,255,255,255,255,255,255,255,243,0,0,0,0
651 DATA 0,0,0,0,5,85,85,85,255,255,0,0,0,0,0,255,255,255,63
652 DATA 255,255,255,255,255,255,255,255,240,0,0,0,0
653 DATA 0,0,0,0,5,85,85,85,85,64,0,0,0,0,15,252,60,255,0,255
654 DATA 255,255,255,255,255,255,255,195,192,0,0,0
656 DATA 0,0,0,0,5,85,85,85,84,0,0,0,0,0,15,240,3,60,255,255
657 DATA 255,255,255,255,255,255,252,0,0,0,0,0
658 DATA 0,0,0,0,5,85,85,85,84,0,0,0,0,0,3,192,12,51,255,255
659 DATA 255,255,255,255,255,252,12,12,0,0,0,0
660 DATA 0,0,0,0,1,85,85,85,80,0,0,0,0,0,0,21,64,0,15,255,255
```

```
890 REM DATA FOR NUMBERS AND LETTERS
900 E$(0)='24444441114444444'
901 E$(1)='24111114444411111'
902 E$(2)='24414444141444414'
903 E$(3)='24414144141444444'
904 E$(4)='24444411114111444'
905 E$(5)='24444144141441444'
906 E$(6)='24444444141441444'
907 E$(7)='24411114111144444'
908 E$(8)='24444444141444444'
909 E$(9)='24444144141444444'
910 E$(10)='24144444114114444
911 E$(11)='24444444141414141
912 E$(12)='24444444111441114
913 E$(13)='24444444111414441
914 E$(14)='24444444141441414
915 E$(15)='24444444141141411
916 E$(16)='24444444111441444
917 E$(17)='24444441141144444
918 E$(18)='24411144444441114
919 E$(19)='24111441111444444
920 E$(20)='24444441141144144
921 E$(21)='24444441111411114
922 E$(22)='44444444111144444411114444
923 E$(23)='24444444111144444
924 E$(24)='24444444111444444
925 E$(25)='24444444141144411
926 E$(26)='24444441411444441
927 E$(27)='24444444141114144
928 E$(28)='24141144141441141
929 E$(29)='24411114444441111
930 E$(30)='24444441111444444
931 E$(31)='24444411111444441
932 E$(32)='44444441111444444411114444
933 E$(33)='24441441141144144
934 E$(34)='24444111144444411
935 E$(35)='24411444141444114
960 RETURN
1010 X=VAL(LEFT$(A$(0),1)):Y=VAL(MID$(A$(0),2,1))
1020 FOR I=0 TO N:A$(I)=RIGHT$(A$(I),LEN(A$(I))-2)
1030 FOR J=0 TO X:FOR K=0 TO Y:T%(I,J,K)=VAL(LEFT$(A$(I),1))
1040 IF LEN(A$(I))=1 THEN 1060
1050 B$=RIGHT$(A$(I),LEN(A$(I))-1):A$(I)=B$
1060 NEXT:NEXT:NEXT:RETURN
1100 FOR I=0 TO Y:FOR J=0 TO X:COLOR T%(N-1,J,I)
1110 SET(X1+J,Y1+I):NEXT:NEXT:RETURN
```

```
    WORLD TIMES     BY R.G.DAVIS  12,7,1984
THE PROGRAM STARTS OFF BY ASKING YOU TO INPUT YOUR
LOCAL TIME THEN IN LINE 12 SUBTRACTS 10 HOURS (FOR
EASTERN TIME). IF YOU LIVE IN S.A LINE 12 WILL READ...
12 Z=H-9:O=M-30
THIS CHANGES THE TIME TO GREENWICH TIME SO AS THE ONLY
LINE THAT WILL EVER HAVE TO BE CHANGED WHERE EVER YOU
LIVE WILL BE LINE 12.
LINE 14 THEN CHECKS FOR ILEGAL TIMES (<ZERO OR >23,59)
LINE 15 THEN DRAWS THE MAP
EXTRA CITIES CAN BE ADDED, AS AN EXAMPLE.....
80 E$="HAWAII":Z=GH-10:O=GM-30:C1=3:C2=2:X3=5:Y3=33:GOSUB 110
TO LOOK AT THIS IN DETAIL
E$="HAWAII"  WRITES HAWAII IN HI RES
Z=GH-10:O=GM-30  SUBTRACTS 10 HOURS 30 MIN FROM G.M.T
C1=3:C2=2  C1 IS CONTRAST COLOUR TO FLASH PIXEL
C2 RESETS PIXEL TO ORIGNAL COLOUR
X3=5:Y3=33  X,Y CORDS TO FLASH PIXEL
```

MACHINE CODE - ASSEMBLER TUTORIAL #2

Last month we considered the AF register, the general purpose register pairs and data transfer operayions. I left you with a problem to solve. Here's my solution:-

Looking in Zak's book I find:-

LD r,n which means:- Load register r with data n.
and

r:	A	B	C	D	E	H	L
	3E	06	0E	16	1E	26	2E

What were the numbers we wanted? They were:-

n:	1	0F	2	0E	3	0C	4

Now the machine code is easy, and may be entered using the 'E' command of your monitor. Let's enter it at 8000H.

E 8000: 3E 01 06 0F 0E 02 16 0E 1E 03 26 0C 2E 04.

Before running this with G 8000 <RET>, you will see that you can't easily tell what is data and what is byte code. Somehow the CPU keeps track but if you left out a byte the computer would race off doing operations you never intended. This perhaps is the greatest problem with machine code.

Let's do the same programme in assembler with changes to allow us to see the register actions. We will use some of the six types of data transfer operations reviewed in the first tutorial.
How may we see the data transfers you ask? The simplest is to finally load the register contents into memory. We could look at those sites by using the 'H' command after running the programme. It is simpler to plan our programme so that we may see the lot in one 'H' block.
We'll do that.

```
                ;       ** DATA TRANSFER DEMO **
                ;
8000    ORG 8000H
                ;
8000    3E01    LD A,1
8002    060F    LD B,0FH
8004    0E02    LD C,2
8006    160E    LD D,0EH
8008    1E03    LD E,3
800A    260C    LD H,0CH
800C    2E04    LD L,4H
                ;The assembler got it right!!!!!
                ;NEXT, REGISTER <- REGISTER  AND
                ;       MEMORY   <- REGISTER.
800E    320A00  LD (0010),A     ;put the value in A (01 at
                                ;present),into memory
                                ;position 000AH
8011    78      LD A,B
8012    320C00  LD (0012),A     ;from B through A to loc'n 12
8015    79      LD A,C
```

SEE NOTE

© John Waters October 1984
All material in Ue Zee News is subject to copyright. Contributed programs are reproduced with permission and on the understanding that they are for the private use of subscibers only. Copyright is retained by the author.

```
8016    32 ØEØØ LD(ØØ14),A      ;From C through A to Loc'N 14
8019    7A      LD A,D
801A    321000  LD (0016),A     ;boring now isn't it !!
801D    7B      LD A,E
801E    321200  LD (0018),A
8021    7C      LD A,H
8022    321400  LD (0020),A
8025    7D      LD A,L
8026    321600  LD (0022),A     ;I told you A is a super-
                                ;market trolley! Must be a
                                ;quicker (not simpler way)
8029            END
```

SEE NOTE

You may not yet been able to get your hands on an assembler so if you have got yourself a monitor you can enter it using the 'E' command.

```
8000: 3E 01 06 0F 0E 02 16 0E 1E 03 26 0C 2E 04 32 0A
8010: 00 78 32 0C 00 79 32 0E 00 7A 32 10 00 7B 32 12
8020: 00 7C 32 14 00 7D 32 16 00 00 00 00 00 00 00 00
```

Run the programme now, using G 8000. The screen will lock up for all that fiddling with the registers is to much for the Monitor. Use your BREAK command in your monitor, and an examination of the beginning of memory should reveal the followig result:-

```
B000: 00 00 00 00 00 00 00 00
B008: 00 00 01 00 0F 00 02 00
B010: 0E 00 03 00 0C 00 04 00
B018: 00 00 00 00 00 00 00 00
```

YES, this is a very simple programme and for some of us, poorly designed and inefficient. However some of you have asked about machine code routines and/or programmes so a good place to start is at the beginning. From this exersise you should learn some of the quirks of assembler coding even if you can't get things to run. Reading and understanding are the hardest.

The routine illustrates the change in order that is required in translating addresses and larger (more than 1 byte) numbers into M/C for the Z80.

Assembler code isn't difficult if kept simple. Like basic, it takes a while to learn the rules that you must adhere to, and the peculiar 'grammar' you use.

BEST OF LUCK AND LOADS OF UNDERSTANDING

)TILL NEXT TIME.)
don't worry if this routine
doesn't run the theory of it
will teach you a lot.....

NOTE

The first few tutorials were written before the VZ Editor Assembler was available so they were done on a System 80. To run this example on your VZ all the numbers marked with an arrow should be changed as follows.

0010 changed to 0B010H
0012 changed to 0B012H
etc.
to
0022 changed to 0B022H

Another Newsletter

There is another newsletter produced for VZ 200 users. It is called "LE' VZ 200 OOP".

It is put out monthly at $1 per issue, the address is as follows.

John C. E. D'Alton
39 Agnes St.
Toowong QLD. 4066

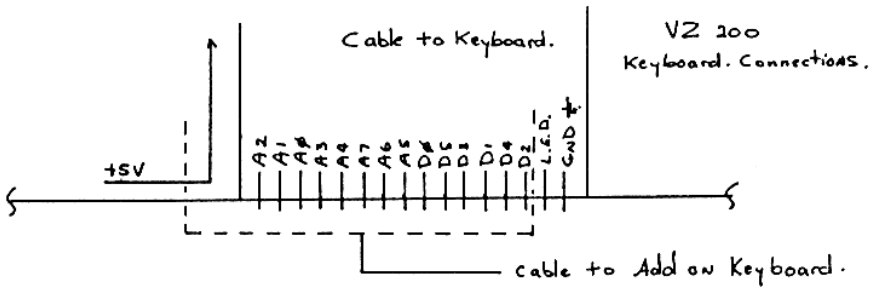
V..Z. 200 KEYBOARD MODIFICATION

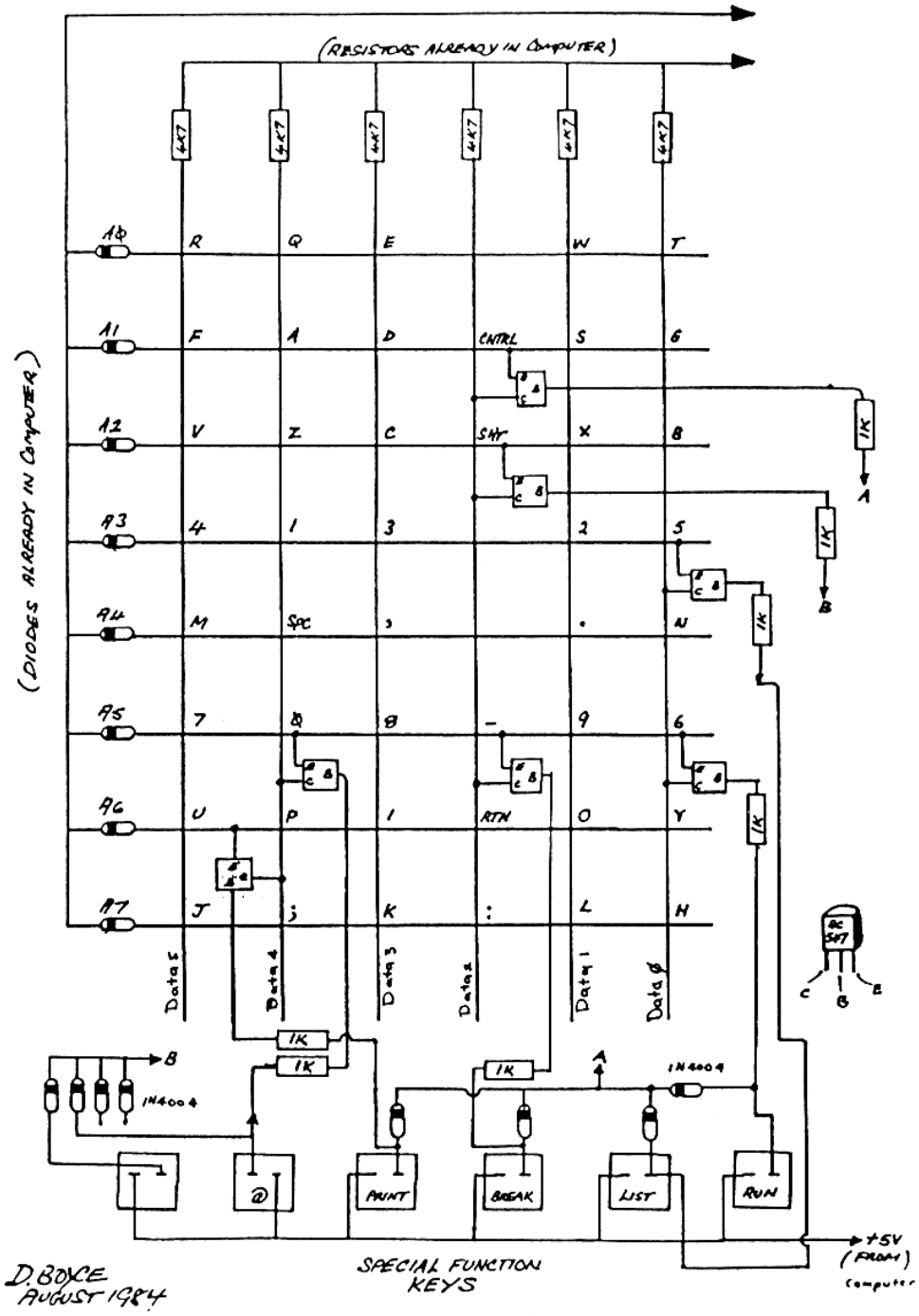
By David Boyce

Although the V.Z. 200 has a full size keyboard, the key's themselves leave a lot to be desired.
It is a relatively simple task to add a keyboard (full-stroke keys) thereby ridding yourself of several problems in one go.
1. Keywobble 2. Lack of feedback. 3. Double and triple keying.
However, it is only fair to point out that attaching a standard keyboard to the V.Z. does have some disadvantages.
1 Lack of legend above and below keys. 2. It's another piece "hanging on". 3. It will most certainly void your warranty. 4. The cable must be brought out of the case, meaning a plug or socket cut-out is needed.
Several other additions to the normal keyboard (see diagram) result in single-function keys such as; PRINT, LIST, BREAK, @ SYMBOL, BACKSPACE, RUN.
Also now possible is two shift keys and a full size space-bar.

CONSTRUCTION

Whilst the actual wiring is straightforward enough, it is time consuming and tedious and is not recommended if you don't like fishing.
The thing that needs most care is attention to the position of each key in relation to the matrix (see diagram).
Make sure that you come off the correct place in the computer i.e. take it from the circuit board end and not the keyboard end of the cable.
Take all precautions, especially if a beginner. When soldering to the various points it is wise to heat your iron then switch off before soldering each point. Be sure to check for bridges and any dry joints.
Incidentally, a word of caution. If you have the SLIGHTEST DOUBT as to your ability to cope, THEN DON'T, because the risk and the responsibility is all yours.
The keyboard modification works - and works well - just takes time.



UNDER SIDE VIEW of P.C. Board.
(components other side).

The Key-board used for the Modification can be bought from Force Electronics in Adelaide ($22.50) It is the 'Micro-Bee' K/Board as advertised by Altronics.